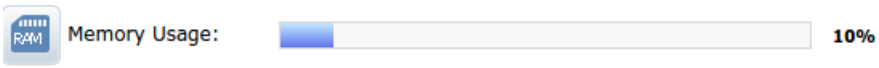
Memory usage insights in FortiOS v5.0

Memory usage on the Fortigate is represented as a single gauge in the GUI or a counter in SNMP, ranging from 0 to 100%.

Even though this counter is easy to read, it needs to be analyzed with deeper attention when reaching high values as it is mixes multiple memory related indicators.

1. One gauge, many memory usages



The 10% memory usage reported in this sample GUI widget (or get sys perf status) is a top-level view of used physical memory.

“diag hard sys memory” gives further details about how the memory is allocated.

```
# diag hard sys mem
             total:       used:       free:  shared:  buffers:   cached:       shm:
Mem:   8350306304  874483712 7475822592          0 117002240 283897856 274530304
Swap:          0          0          0
MemTotal:        8154596 kB
MemFree:         7300608 kB
MemShared:             0 kB
Buffers:          114260 kB
Cached:           277244 kB
SwapCached:            0 kB
Active:           125748 kB
Inactive:         265840 kB
HighTotal:             0 kB
HighFree:              0 kB
LowTotal:        8154596 kB
LowFree:         7300608 kB
SwapTotal:             0 kB
SwapFree:              0 kB
```

It is interesting to note that

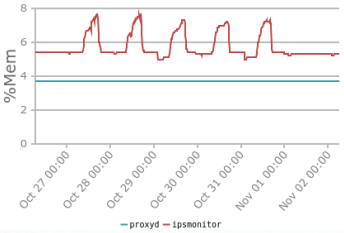
- The value displayed in the GUI widget is the ratio used / total
- Used memory is dispatched between buffers, cached, shared memory and the non-system area (aka user space)

2. User / application memory space

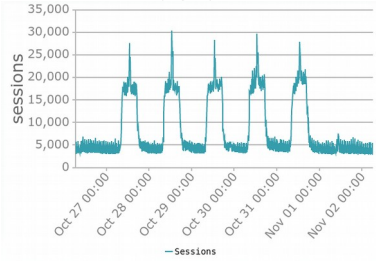
FortiOS applications, such as the UTM components, web and CLI based interfaces, run in the user space (non-system area). Application memory usage can be viewed using

```
# diag sys top-summary -sort=mem

   CPU [|                                                  ]   4.3%
   Mem [||||                                               ]  10.0%   807M/7963M
   Processes: 20 (running=1 sleeping=99)

   PID      RSS   CPU% ^MEM%   FDS     TIME+  NAME
 * 97      150M    0.0  1.9  6726  00:00.88  proxyd [x8]
   93       90M    0.0  1.1    14  00:04.82  reportd
   957      30M    0.0  0.4    12  00:00.80  pyfcgid [x4]
   107      27M    0.0  0.3    30  00:40.28  hasync
   116      26M    3.8  0.3    15  00:01.60  sshd [x4]
   52       25M    0.0  0.3    13  00:05.96  cmdbsvr
   76       25M    0.0  0.3    18  00:05.51  httpsd [x4]
   73       24M    0.0  0.3    28  00:00.80  miglogd [x2]
   79       22M    0.0  0.3    18  00:00.40  ipsmonitor [x2]
   94       22M    0.0  0.3    30  00:00.24  sslvpnd [x4]
   125      15M    0.0  0.2    16  00:00.00  fgfmd
   98       14M    0.0  0.2    31  01:12.68  iked
   126      14M    0.0  0.2    24  00:00.20  cw_acd
   144      14M    0.0  0.2    14  00:01.16  updated
   122      13M    0.0  0.2    29  00:36.21  dnsproxy
```

This command shows the memory allocated for each process tree (parent and children), as amount of memory held in RAM (RSS) and its ratio over the total memory (MEM%).



Application memory usage fluctuates with the process activity. On a busy UTM system, it is expected to see IPS engine (ipsmonitor) or the transparent proxies (proxyd) owning a lot of memory (up to 50~60%). When activity decreases, memory gets released.



In this example, showing “diag sys top-summary” over 1 week, ipsmonitor allocates memory during the day and releases it every night, when activity is quieter.

Note that Shared Memory is not accounted here.

3. Kernel buffers

The kernel buffers are allocated for all system related tasks. These are mainly for network buffers, filesystem structure buffers, and generic usage fixed-size buffers.

Detailed listing of the kernel buffers is available with

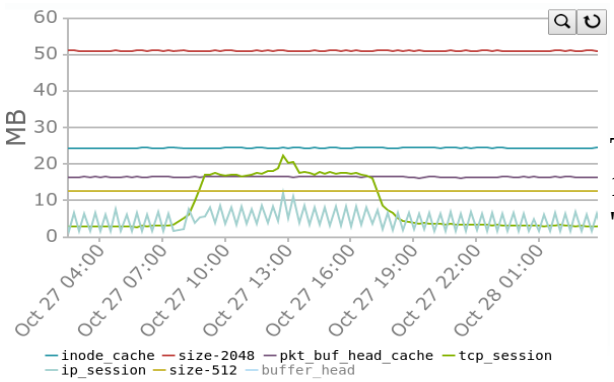
```
# diag hard sys slab
```

```
slabinfo - version: 1.1 (SMP)
sctp_session           0      0   1152    0    0    2 0 :   60   30
tcp_session          442   3052   1152   98  436    2 338 :   60   30
ip_session          5859   7749   1088  867 1107    2 240 :   60   30
tcp_open_request     634    760    192   38   38    1 0 :  252  126
inet_peer_cache      504    630    128   18   21    1 3 :  252  126
ip_dst_cache        3230   3912    320  302  326    1 24 :  124   62
ip_fib_hash          448    448     32    4    4    1 0 :  252  126
arp_cache            984   1110    256   72   74    1 2 :  252  126
mnt_cache            120    120    128    4    4    1 0 :  252  126
inode_cache        12013  12075    768 2415 2415    1 0 :  124   62
dentry_cache       12280  12280    192  614  614    1 0 :  252  126
buffer_head        41740  41740    192 2087 2087    1 0 :  252  126
fs_cache             354    354     64    6    6    1 0 :  252  126
size-2048(DMA)       540    660   2048  274  330    1 56 :   60   30
size-2048            500    500   2048  250  250    1 0 :   60   30
size-1024(DMA)       258    444   1024   88  111    1 23 :  124   62
size-1024            820    820   1024  205  205    1 0 :  124   62
…
```

Kernel buffers are using a “slab” memory management mechanism, where each buffer has a fixed size (1st column), that is adjusted to store the underlying kernel object.

The kernel can then allocate the number of buffer objects required to store the related object type (3rd column).

In the above example, FortiOS has allocated 1152 buffers of 442 bytes each to store objects of type 'tcp_session', which represents a total of 509184 bytes, or approximately 500kB.



This example shows 'diag hard sys slab' over 1 day. We can note the firewall activity as 'tcp_session' between 09:00 and 18:00

3. Cache memory

This memory area is mainly used for disk I/O buffering. It caches program/data files instead of reloading them from the slow storage device.

It comprises two sections: Active + Inactive.

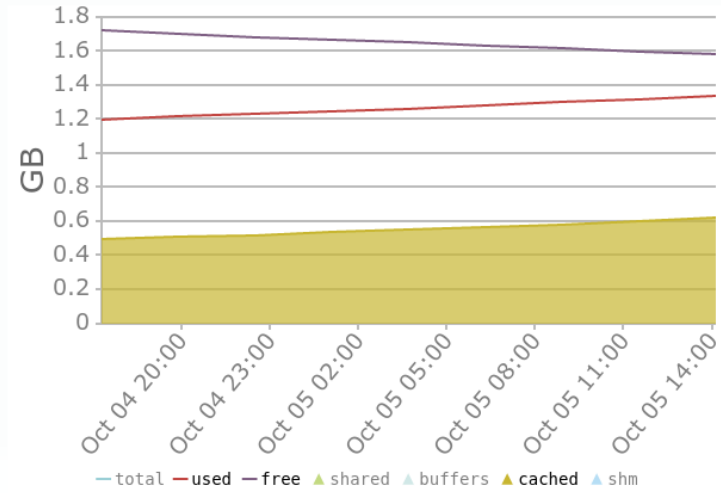
Active memory is considered as busy memory. It contains data related to files that are currently open.

Inactive memory is almost free memory, albeit accounted as “used”. It contains data that are no longer being accessed by processes, such as recently closed files. FortiOS considers it is judicious to keep this data held in RAM, so that if it needs to be accessed again, no time is spent in accessing the storage device.

As long as the system is not under memory pressure, the inactive cache will slowly grow over time if disk related features are enabled.

However, when memory usage reaches ~70%, the system will reclaim memory from the inactive cache and stabilize at this level.

```
# diag hard sys mem
             total:     used:      free:  shared:  buffers:   cached:      shm:
Mem:  8350306304 874483712 7475822592        0 117002240 283897856 274530304
Swap:        0        0        0
MemTotal:       8154596 kB
MemFree:        7300608 kB
MemShared:            0 kB
Buffers:         114260 kB
Cached:          277244 kB
SwapCached:           0 kB
Active:          125748 kB
Inactive:        265840 kB
HighTotal:            0 kB
HighFree:             0 kB
LowTotal:       8154596 kB
LowFree:        7300608 kB
SwapTotal:            0 kB
SwapFree:             0 kB
```



4. Shared Memory

This memory area primary purpose is to allow fast passing of data between processes. Shared memory blocks are owned by the process who allocated them, but they can be used by other processes.

Shm is allocated/released on demand and is expected to vary along with the system load.

```
# diag hard sys mem
             total:     used:      free:  shared:  buffers:   cached:      shm:
Mem:  8350306304 874483712 7475822592        0 117002240 283897856 274530304
Swap:         0         0         0
```

5. memory pattern

Monitoring memory usage of the Fortigate shows continuous evolution around an average value throughout the day, with slow increases and sudden drops.

This typical pattern shape is related to FortiOS memory manager, who allocates memory pages on demand (slow increase), and release them in groups (sudden drop).